Jerzy Konorski
Jakub Grochowski
Wydział Elektroniki, Telekomunikacji i Informatyki
Politechnika Gdańska
ul. Gabriela Narutowicza 11/12, 80-233 Gdańsk
Adres e-mail: jekon@eti.pg.edu.pl

KTiT 2019
Wrocław, 26-28 czerwca 2019

DOI: 10.15199/59.2019.7.13

OBRONA SIECI INFORMACJOCENTRYCZNEJ PRZED ZATRUWANIEM TREŚCI PRZEZ NIEZAUFANYCH WYDAWCÓW Z UŻYCIEM MODELU INFEKCJI W GRAFACH

DEFENSE OF AN INFORMATION-CENTRIC NETWORK AGAINST CONTENT POISONING BY COMPROMISED PUBLISHERS USING A MODEL OF INFECTION ON GRAPHS

Streszczenie: **Sieci informacjocentryczne narażone są na ataki zatruwania treści przez intruza, który przejął klucz prywatny wydawcy treści. Efektem jest podmiana treści oryginalnych na zatrute. W pracy zaproponowano model ataku opierający się na analogii z procesami infekcji w grafach i przeanalizowano prosty mechanizm obronny. Symulacje przeprowadzone w sieciach informacjocentrycznych o topologiach bezskalowych oraz losowych wykazały potencjalną efektywność proponowanego mechanizmu obronnego oraz wpływu liczby węzłów sąsiednich.**

Abstract: **Information-centric networks are susceptible to content poisoning attacks by compromised publishers, resulting in replacement of original contents by poisoned ones. This paper proposes an attack model based on infection processes in graphs and analyses a simple defense mechanism. Simulations performed in information-centric networks of scale-free and random topologies have shown the potential effectiveness of the proposed defense mechanism and the influence of the average number of a network node's neighbors.**

Słowa kluczowe: **sieci informacjocentryczne, Named Data Networking, zatruwanie treści, model infekcji**

Keywords: **Information Centric Networking, Named Data Networking, content poisoning attack, infection model**

1. WSTĘP

Sieci informacjocentryczne (ang. *Information Centric Networking*, ICN) są nową i wciąż rozwijaną technologią której celem jest rozwiązanie problemu nowoczesnego Internetu, jakim jest nadmierne zużycie zasobów sieciowych (np. pasma) związane z odległością między serwerami oferującymi w sieci pewne usługi lub treści a użytkownikami nimi zainteresowanymi. Podejściem stosowanym w architekturach ICN jest rozpowszechnianie kopii dostępnych treści do pamięci podręcznych istniejących w sieci urządzeń (węzłów) – dzięki temu prawdopodobnym jest, że użytkownik żądający od sieci udostępnienia pewnej treści (np. pliku) otrzyma ją z repozytorium w pamięci podręcznej pobliskiego węzła bez angażowania centralnych serwerów treści i związanego z tym obciążenia zasobów sieciowych [4][11].

Zatruwanie treści (ang. *content poisoning attack,* CPA) jest typem ataku, w którym intruz podmienia pewną dostępną treść na tak zwaną treść zatrutą. Zatruta treść posiada takie same metadane jak oryginalna treść poprawna, lecz różni się pod względem zawartości – powodując, że użytkownik otrzymuje treść fałszywą i być może szkodliwą. Sieci ICN są szczególnie narażone na atak przez zatruwanie treści, co wynika z samej jej zasady działania, zakładającej rozpowszechnianie treści do pamięci podręcznych węzłów i co za tym idzie, szybką ich propagację w całej sieci. Propagację zatrutych treści można ograniczyć, a ich szkodliwość dla użytkownika wyeliminować, poprzez umieszczenie w metadanych podpisu cyfrowego wydawcy umożliwiającego weryfikację poprawności treści [4]. Jednakże weryfikacja treści przez węzeł jest rozwiązaniem mało praktycznym z uwagi na duży narzut czasowy i komunikacyjny [7]. Jednocześnie nie zapobiega ona atakom przejęcia prywatnego klucza wydawcy treści przez intruza (atak taki można określić angielskim terminem *content poisoning by compromised publisher*, $(CP)^2$). Jako że technologia ICN jest ciągle na wczesnym etapie rozwoju, dotychczasowe propozycje mechanizmów obrony przed atakiem $(CP)^2$ są nieliczne i nie wchodzą w skład standardowych architektur ICN [7].

W niniejszej pracy zaproponowano mechanizm obrony przed atakiem $(CP)^2$ poprzez ograniczenie propagacji zatrutej treści w sieci. Dla każdej nowej treści zapewnia on istnienie niepustego zbioru węzłów, które ostatecznie oznakują tę treść jako niezaufaną (przy domniemaniu zatrucia). Dotyczy to zarówno treści zatrutych, jak i poprawnych (jako nieodróżnialnych), stąd zbiór ten powinien mieć kontrolowany rozmiar i na ogół być różny dla różnych treści, tak by dostępność treści oznakowanych jako zaufane (przy domniemaniu poprawności) była względnie równomierna w sieci. Symulacje przeprowadzone w środowisku ICN pozwoliły zbadać wpływ parametrów mechanizmu oraz topologii sieci – w szczególności średniej liczby węzłów sąsiednich węzła – na jego efektywność.

2. MODEL

2.1. Sieć informacjocentryczna

Za jedną z bardziej obiecujących architektur ICN uznaje się Named Data Networking (NDN) [9]. W architekturze NDN każda dostępna treść podzielona jest na jeden lub kilka fragmentów, zwanych *obiektami danych*. Każdy obiekt danych posiada unikatową nazwę, która umożliwia jego identyfikację. Stosowany w NDN schemat nazewnictwa danych zakłada zrozumiałość dla

człowieka oraz członowość nazw. Człony nazwy oddzielone są znakiem "/", przez co nazwa przypomina ścieżkę dostępu, np. "/ndn/youtube/bob/vid50/13". Interesujące z punktu widzenia NDN podmioty komunikacji należeć mogą do jednego z trzech rodzajów: *wydawca* (ang. *Publisher*) odpowiada za przechowywanie obiektów danych i udostępnianie ich, *subskrybent* (ang. *Subscriber*) jest użytkownikiem zainteresowanym określoną treścią i generuje żądania udostępnienia określonych obiektów danych, zaś *węzeł* (ang. *Content Router*) odpowiada za komunikację w sieci poprzez przekierowywanie otrzymywanych wiadomości w stronę odpowiednich, widocznych dla niego sąsiadów – innych węzłów, wydawców lub subskrybentów. NDN zakłada, że wiadomości przekierowywane są pojedynczymi skokami (tj. metodą *hop-by-hop*). Wyróżnia się dwa typy wiadomości: *Interest* – zawiera żądanie dotyczące udostępnienia określonego obiektu danych i wysyłana jest od subskrybenta w kierunku odpowiedniego wydawcy, oraz *Data* – zawiera obiekt danych i przesyłana jest jako odpowiedź na wiadomość *Interest* [9].

2.2. Działanie węzła

Każdy z węzłów utrzymuje trzy struktury danych niezbędne do poprawnego działania sieci [9]. Tablica kierowania (ang. *Forwarding Information Base*, FIB) zawiera informacje dotyczące przekierowywania wiadomości *Interest* – każdemu z dostępnych w sieci obiektów danych przypisuje ona widocznego sąsiada, w stronę którego należy przekierować wiadomość, aby trafiła ona do odpowiedniego wydawcy. Tablica żądań (ang. *Pending Interest Table*, PIT) zawiera informacje dotyczące otrzymanych przez węzeł, lecz jeszcze niespełnionych żądań. W przypadku, gdy węzeł przekierowuje wiadomość *Interest* w kierunku wskazanym przez FIB, do PIT dopisany zostaje wpis zawierający informacje o nazwie żądanego obiektu danych oraz sąsiada, który wysłał wiadomość *Interest* do węzła. W przypadku, gdy węzeł otrzymuje wiadomość *Data*, zostaje ona przekierowana w kierunku wskazanym przez wpisy PIT – następnie odpowiednie żądania zostają uznane za spełnione i wpisy te usunięte są z PIT. Trzecią ze struktur jest pamięć podręczna węzła (ang. *Content Store*, CS). Obiekty danych zawarte w wiadomościach *Data* przekierowywanych przez węzeł mogą zostać zapisane w CS. Po otrzymaniu wiadomości *Interest* dotyczącej obiektu znajdującego się w jego CS węzeł nie przekazuje jej dalej, lecz odpowiada wiadomością *Data* zawierającą obiekt z CS – zmniejsza to obciążenie zasobów sieci poprzez skrócenie tras przesyłania obiektów danych [11].

2.3. Uaktualnianie treści

Wydawcy mogą wprowadzać do sieci nowe wersje obiektów danych. W metadanych opisujących dostępne obiekty danych znajduje się pole *Freshness*, tj. wartość liczbowa opisująca sugerowany przez wydawcę termin aktualności obiektu – w przypadku dwóch obiektów o tej samej nazwie lecz różnych wartościach *Freshness*, wartość wyższa oznacza wersję bardziej aktualną. Zakładamy, że subskrybent chce otrzymać najnowsze wersje dostępnych treści, zadaniem węzłów jest więc przechowywanie w CS wersji obiektów o najwyższych zaobserwowanych wartościach *Freshness*. Poprzednie wersje również mogą być przechowywane w CS, jeżeli nie jest ona przepełniona. Otrzymanie nowej wersji obiektu danych oznacza, że nie wszystkie węzły w sieci mogą już posiadać najnowszą wersję obiektu w swoich CS, tym samym przesyłają wiadomości *Data* zawierające obiekty przedawnione. Wobec tego węzeł, który zapisał w CS nową wersję, propaguje ją poprzez wysłanie wiadomości *Data* do wszystkich węzłów sąsiednich. Umożliwia to szybkie uaktualnienie zawartości CS węzłów w sieci, jednakże umożliwia również atak $(CP)^2$. Przyjmujemy, że atak dotyczy pojedynczego obiektu danych (dla różnych obiektów można go więc analizować niezależnie), zaś jego celem jest propagacja w sieci zatrutego obiektu danych jako nowej wersji. Dla skutecznego wykonania ataku $(CP)^2$ wystarczy, że po przejęciu klucza prywatnego wydawcy intruz wyśle do węzła zatruty obiekt danych o aktualnie najwyższej wartości *Freshness*, tak by obiekt ten został zapisany w CS, a następnie jako wiadomość *Data* rozesłany do pozostałych węzłów sieci, zastępując poprawną, niezatrutą wersję obiektu [3]. W architekturze NDN każdy nowy obiekt danych jest bowiem propagowany przez węzły w sieci, niezależnie od tego czy jest zatruty.

2.4. Mechanizm obronny

Proponowany mechanizm obrony przed $(CP)^2$ zakłada, że nowe wersje obiektów danych są rozsyłane w wiadomościach *Data* jedynie wówczas, gdy są oznakowane przez węzeł jako *zaufane*. Nowe wersje obiektów danych otrzymane od wydawcy są znakowane jako zaufane, zaś otrzymane od innych węzłów są znakowane jako zaufane bądź *niezaufane* w zależności od informacji od węzłów sąsiednich o zaufaniu lub też braku zaufania do danej wersji obiektu danych. (Wzajemne odpytywanie w celu ustalenia oryginalności danych jest rozwiązaniem często spotykanym w rozproszonych systemach dostarczania treści, por. np. protokół LCAP zaimplementowany w systemie LOCKSS [8].) Otrzymanie przez węzeł *i* od węzła sąsiedniego *j* wiadomości *Data* zawierającej nową wersję obiektu danych jest jednoznaczne z tym, że węzeł *j* oznakował tę wersję jako zaufaną. Jeśli dostatecznie wiele węzłów sąsiednich węzła *i* jest w takiej sytuacji, to nowa wersja zostaje przez węzeł *i* oznakowana jako zaufana.

Do informowania o braku zaufania służy wiadomość *Suspicion*, dotycząca niezaufanej wersji obiektu danych. W przypadku gdy węzeł *j* nie przechowuje nowej wersji w CS, bądź uprzednio oznakował ją jako niezaufaną, lecz jeden z jego węzłów sąsiednich *k* nadesłał odpowiednią wiadomość *Data*, węzeł *j* staje się *podejrzliwy* wobec tej wersji obiektu danych i wysyła wiadomości *Suspicion* do wszystkich swoich węzłów sąsiednich z wyjątkiem *k*. Tak więc dowolny węzeł *i* otrzymuje od swojego węzła sąsiedniego *j* wiadomość *Suspicion* dotyczącą nowej wersji obiektu danych, gdy *j* nie oznakował tej wersji jako zaufanej, lecz posiada przynajmniej jeden węzeł sąsiedni *k* różny od *i*, który to uczynił. Otrzymanie przez węzeł *i* wiadomości *Suspicion* od wystarczająco wielu węzłów sąsiednich powoduje, że węzeł *i* bezpowrotnie oznakowuje tę wersję obiektu danych jako niezaufaną (nie mogą tego zmienić przyszłe wiadomości *Data*).

MOST WIEDZY Pobrano z mostwiedzy.pl

2.5. Model infekcji

Posługując się analogią procesów infekcji w grafach (gdzie wierzchołkami grafu są węzły sieci NDN, zaś krawędzie reprezentują bezpośrednie sąsiedztwo między węzłami), możemy powiedzieć, że węzeł ma możliwość *infekowania* węzłów sąsiednich nową wersją obiektu danych. Propagacja nowej wersji każdego obiektu danych w sieci jest więc reprezentowana przez proces infekcji zgodnie z pewnym modelem infekcji.

Wykorzystywanym tutaj modelem infekcji jest nieznacznie zmodyfikowany model SIR [6]. Model ten zakłada, że każdy wierzchołek może się znajdować w jednym z trzech stanów. Stan *podatny* (ang. *Susceptible*) opisuje węzeł, który dotąd nie otrzymał nowej wersji obiektu danych, lub nie oznakował jej jako zaufanej, choć może to uczynić w przyszłości. Stan *zainfekowany* (ang. *Infected*) opisuje węzeł, który posiada nową wersję obiektu danych w CS i jest ona tam oznakowana jako zaufana. Natomiast stan *wyleczony* (ang. *Recovered*) opisuje węzeł, który w przeszłości oznakował nową wersję obiektu jako zaufaną, lecz obecnie jest ona tam bezpowrotnie oznakowana jako niezaufana. Widać, że w odróżnieniu od klasycznego modelu SIR węzły wyleczone stają się następnie *odporne*: węzeł wyleczony nie może zostać ponownie zainfekowany.

Zgodnie z opisanym mechanizmem obronnym węzły nie infekują się bezwarunkowo: węzeł podatny zmieni stan na zainfekowany, tj. oznakuje nową wersję obiektu danych jako zaufaną, przy spełnieniu warunku:

$$\frac{I_i}{N_i} > \xi \wedge \frac{S_i}{N_i} \leq \eta, \tag{1}$$

gdzie N_i oznacza liczbę węzłów sąsiednich węzła i, I_i i S_i oznaczają odpowiednio liczby węzłów sąsiednich węzła i w stanie zainfekowanym oraz podejrzliwych, zaś $\xi, \eta \in [0, 1]$ są ustalonymi parametrami proponowanego mechanizmu obronnego, określającymi odpowiednio podatność węzła na infekcję i wyleczenie. Jest to więc zmodyfikowany wariant procesu infekcji typu *bootstrap percolation*, por. np. [2]. Podobnie, stan węzła zainfekowanego zmieni się na wyleczony, tj. nowa wersja obiektu danych zostanie bezpowrotnie oznakowana jako niezaufana, przy spełnieniu warunku:

$$\frac{S_i}{N_i} > \eta, \tag{2}$$

gdzie, analogicznie do (1), N_i i S_i oznaczają odpowiednio liczbę wszystkich i podejrzliwych węzłów sąsiednich węzła i, zaś parametr η został zdefiniowany wyżej. Z uwagi na założoną odporność węzłów wyleczonych, mamy do czynienia z wariantem procesu infekcji z uodpornianiem [10]. W przypadku zmiany stanu węzeł i wysyła wiadomości do węzłów sąsiednich: *Data*, gdy nowym stanem jest stan zainfekowany, bądź *Suspicion*, gdy nowym stanem jest stan wyleczony oraz co najmniej jeden z węzłów sąsiednich jest zainfekowany. Węzły sąsiednie uaktualniają informacje o stanie własnych sąsiadów i w rezultacie mogą również zmienić stan.

Ponieważ model infekcji dotyczy zarówno zatrutych jak i poprawnych obiektów danych (które w trakcie ataku $(CP)^2$ są nieodróżnialne), niepożądane jest, by mechanizm obronny całkowicie wykluczał nieograniczoną infekcję węzłów – zablokowałoby to propagację nowych wersji poprawnych obiektów danych. Z drugiej strony nieograniczona infekcja umożliwiałaby łatwy dostęp do poprawnych obiektów, lecz powodowałaby ryzyko pełnej podmiany poprawnych obiektów przez wersje zatrute. Istotne jest więc, by rezultatem mechanizmu obronnego był niepusty, lecz zarazem i niepełny zbiór węzłów X, które oznakują nową wersję obiektu danych jako zaufaną. Implikuje to pewien minimalny warunek bezpieczeństwa: dla każdego zatrutego obiektu danych istnieje niepusty zbiór węzłów, które oznakują jego nową wersję jako niezaufaną oraz dla każdego poprawnego obiektu danych istnieje niepusty zbiór węzłów, które oznakują jego nową wersję jako zaufaną. Zbiór X nie powinien być jednakowy dla wszystkich obiektów danych, aby zapewnić względnie równomierną dostępność poprawnych obiektów danych, ani łatwo przewidywalny dla intruza (np. może zależeć od zwykle przypadkowej kolejności uaktualniania stanów węzłów).

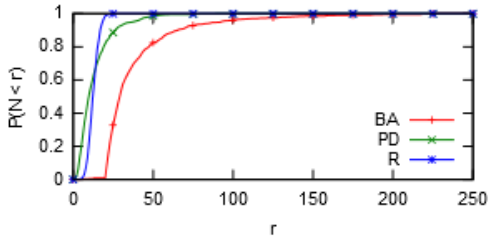
3. SYMULACJA I WYNIKI

Symulacje przeprowadzono za pomocą środowiska symulacyjnego OMNeT++ w wersji 4.6. Liczba węzłów wynosiła n = 1000, zaś łączny czas symulacji wynosił 750000 s i podzielony został na 2500 szczelin czasowych o długości 300 s każda. Na potrzeby symulacji stworzono dwa typy modułów: moduł *Node* odzwierciedla działanie pojedynczego węzła poprzez zgodne z modelem zarządzanie pamięcią podręczną (CS) oraz reagowanie na otrzymane od węzłów sąsiednich wiadomości typu *Data*, *Interest* bądź *Suspicion*, różnicowane poprzez wartość istniejącego w klasie cMessage atrybutu *kind*. Natomiast moduł *Collect* zbiera informacje dotyczące stanów każdego z modułów *Node* oraz odpowiada za infekowanie ich zewnętrznie: scenariusz symulacji zakłada, że w każdej szczelinie czasowej jeden losowo wybrany węzeł w stanie podatnym zostaje zainfekowany zewnętrznie przez nową wersję zatrutego obiektu danych, tzn. jego stan zmienia się na zainfekowany niezależnie od (1) i (2). Symulacje przeprowadzono dla różnych wartości parametrów ξ i η oraz różnych metod generowania topologii sieci. Porównano wyniki otrzymane dla sieci o topologiach grafów bezskalowych wygenerowanych metodami: Barabasi-Albert (BA) [1] oraz probabilistycznej duplikacji (PD) [5], a także dla sieci o topologii grafu przypadkowego, w dalszym ciągu oznaczanej jako R.

Metoda BA rozpoczyna od m_0 wierzchołków, następnie dla każdego kolejnego dodawanego wierzchołka (aż do osiągnięcia liczby n) generuje m nowych krawędzi z losowo wybranymi wierzchołkami już istniejącymi. Metoda PD rozpoczyna od spójnego grafu przypadkowego z n_0 wierzchołkami i minimum e_0 krawędziami. Następnie dla każdego kolejnego dodawanego wierzchołka (aż do osiągnięcia liczby n) wybierany jest losowo jeden z istniejących już wierzchołków i tworzone są krawędzie łączące nowy wierzchołek z tym wierzchołkiem oraz, z prawdopodobieństwem p, z każdym z jego sąsiadów. W topologii R najpierw tworzy się spójne losowe drzewo z n wierzchołkami, następnie między wierzchołkami generowane są losowo krawędzie, tak by łączna liczba krawędzi była taka sama, jak dla PD.

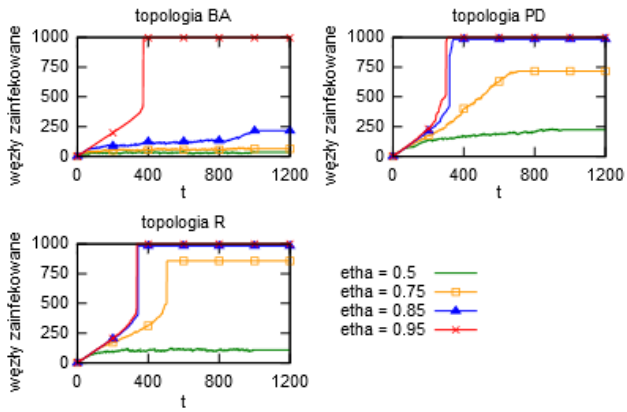
MOST WIEDZY Pobrano z mostwiedzy.pl

Rozkłady prawdopodobieństwa liczebności sąsiadów węzłów N_i widoczne są na Rys. 1. Wartości parametrów dla BA: $m_0 = 50$, $m = 20$, dla PD: $n_0 = 20$, $e_0 = 10$, $p = 0{,}5$. Zauważmy, że metoda BA gwarantuje większości węzłów minimum m węzłów sąsiednich, w rezultacie średnia liczba węzłów sąsiednich jest większa niż dla metody PD.



Rys. 1. Rozkład liczby węzłów sąsiednich

Rys. 2 pokazuje przykładowe trajektorie procesu infekcji w sieci informacjocentrycznej, tzn. zmiany w czasie liczby węzłów zainfekowanych zatrutym obiektem danych, dla rozważanych topologii BA, PD i RD. Trajektorie przedstawiono dla różnych wartości parametru η przy ustalonej wartości parametru $\xi = 0{,}5$.



Rys. 2. Trajektorie procesu infekcji

Dla dostatecznie wysokich wartości parametru η można zauważyć znane w literaturze, zwłaszcza w odniesieniu do grafów bezskalowych, zjawisko *epidemii* [2][9][10]: zewnętrzne infekowanie kolejnych węzłów w pewnym momencie powoduje szybki wzrost liczby zainfekowanych węzłów, ostatecznie obejmującej wszystkie węzły. Dla mniejszych wartości parametru η po wystąpieniu epidemii liczba zainfekowanych węzłów stabilizuje się, pozostawiając niepusty zbiór węzłów X w stanie wyleczonym, które, jako odporne, nie mogą zostać ponownie zainfekowane; niekiedy stabilizacja następuje również bez zjawiska epidemii.

4. WNIOSKI

Topologia R charakteryzuje się lżejszym niż topologia PD ogonem rozkładu liczby węzłów sąsiednich – spowalnia to proces infekcji, lecz dla dużych wartości parametru nie chroni przed zjawiskiem epidemii. Rozkłady liczby węzłów sąsiednich PD i BA są podobne, ze znacząco większą wartością oczekiwaną dla BA. Teoretycznie może to mieć to zarówno pozytywny jak i negatywny wpływ na obronę przed atakiem $(CP)^2$, gdyż zwiększa prawdopodobieństwo, że dany węzeł będzie posiadał odpowiednią liczbę sąsiadów tak podejrzliwych, jak i zainfekowanych. Wyniki pokazują, że pozytywny wpływ jest silniejszy: zjawisko epidemii występuje tylko dla dostatecznie wysokich wartości parametru η. Jednocześnie zbyt niska wartość tego parametru powoduje, że zbiór węzłów zainfekowanych jest bardzo nieliczny, co wprawdzie pozwala na efektywne powstrzymywanie ataku $(CP)^2$, zarazem jednak pogarsza propagację i dostępność poprawnych obiektów danych. Udowodniono więc potencjalną efektywność proponowanego mechanizmu obronnego, jednak wyzwaniem pozostaje jego systematyczna konfiguracja w zależności od topologii sieci NDN, jak też od szczegółów implementacji propagacji nowych treści pomiędzy węzłami.

PODZIĘKOWANIE

Praca sfinansowana przez Narodowe Centrum Nauki (umowa UMO-2016/21/B/ST6/03146).

LITERATURA

[1] Barabási Albert-Laszlo, Albert T. R.. 1999. „Emergence of scaling in random networks. ” *Science* 286 5439 (1999): 509-12.

[2] Chalupa J., Leath P. L., Reich G. R. 1979. „Bootstrap percolation on a Bethe lattice”. *Journal of Physics C: Solid State Physics*, 12 (1).

[3] Ghali Cesar, Tsudik Gene, Uzun Ersin. 2014. „Needle in a haystack: Mitigating content poisoning in named-data networking”. *Proc. NDSS Workshop on Security of Emerging Networking Technologies*.

[4] Ghodsi A., Shenker S., Koponen T., Singla A., Raghavan B., Wilcox J.. 2011. „Information-centric networking: seeing the forest for the trees.” *Proc. 10th ACM Workshop on Hot Topics in Networks*.

[5] Graham Fan Chung, Linyuan Lu, Dewey Gregory T., Galas David J.. 2003. „Duplication models for biological networks”. *J. of Computational Biology* 10, 5 (2003): 677-87.

[6] Newman Mark E. J. 2010. „Networks: An Introduction”. Oxford University Press.

[7] Nguyen Tan N., Marchal Xavier, Doyen Guillaume, Cholez Thibault, Cogranne Rémi. 2017. „Content poisoning in Named Data Networking: Comprehensive characterization of real deployment.” *IFIP/IEEE Symposium on Integrated Network and Service Management*.

[8] Rosenthal David SH. „LOCKSS: Lots of copies keep stuff safe.” *NIST Digital Preservation Interoperability Framework Workshop*. 2010.

[9] Saxena Divya, Raychoudhury Vaskar, Suri Neeraj, Becker Christian, Jiannong Cao. 2016. „Named Data Networking: A survey”. *Computer Science Review* 19 (2016): 15-55.

[10] Zanette Damian H., Kuperman Marcelo. 2002. „Effects of immunization in small-world epidemics.” *Physica A* 309 (2002): 445-452.

[11] Zhang Guoqiang, Yang Li, Tao Lin. 2013. „Caching in information centric networking: A survey.” *Computer Networks* 57, 3128-3141.

MOST WIEDZY Pobrano z mostwiedzy.pl